Tel-Hai academic college

Upper Galil - Israel

Computer science department

**Particle Swarm Optimization - Research On Converging Islands Approach**



Submitted by : Orr Shushan Levi 201183597

Guided and supervised by : Dr. Ofer.M.Shir

**Introduction**

Particle swarm optimization is a method in computational intelligence that optimizes a problem by iteratively improving a candidate solution, with consideration to the quality of the solution.

Alongside with Ant Colony optimization method, both PSO (Particle swarm optimization) and ACO (Ant Colony optimization) are optimization methods based on observation of natural behaviour in animals.

The project emerged from the idea that birds can flock in many different ways.

PSO works in the following way: particles moves around the search-space according to a mathematical formula that will be presented later.

Each particle movement is influenced by it’s local best known position and the swarm leader best position.

The following method creates a representation of birds flocking in nature in search of food or a place to rest.

When taught about PSO i remembered that i saw different groups of birds flocking away from each other and suddenly converging into one big group.

I have asked my professor Dr. Shir at the end of the class if this behavior had been already implemented in the PSO models.

With no information of such model found, we conducted to research the efficiency of the model described.

**The main goal:**

Our main goal was building a software that represent the following model of behaviour:

A swarm of particle will be initialized into K different groups over the search space.

Each group will have its own leader which holds the best candidate solution found so far among all of the particles in it’s group.

The method described above is known as PSO-Island model.

On top of the described model, our goal was to add another layer, a layer of conversion between each group, so that the end result will be one big group of particles.

In the course “computational intelligence” our language of programing was python.

Python is a popular language among developers and have open source features.

Therefore i concluded that using python will benefit any further growth of the project.

Dew to the nature of investigating the efficiency of a new optimization model, the end result could be disappointing.

Our sub goal was to test the converging island model to determine its efficiency.

**Background:**

PSO developed by Kennedy and Eberhart in 1995.

PSO consists of a swarm of particles.Each particle resides at a position in the search space.

The fitness(score) of each particle represents the quality of his position (candidate solution).

Particles travel throughout the search space with a certain velocity.

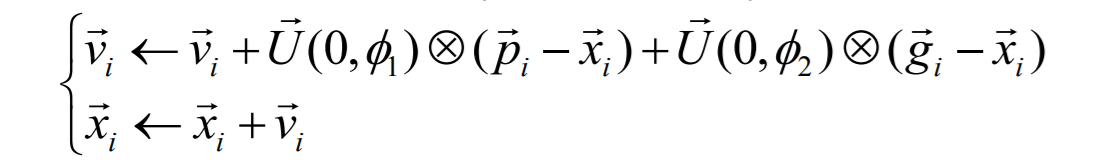
The velocity (both direction and speed) of each particle is influenced by its own best position found so far and the best solution that was found so far by its neighbors.

By the course of time the swarm will converge to optimal positions.

The classic PSO Algorithm:

* Randomly initialize particles positions and velocities
* While stopping criterion is not met do:
* For each particle i:
* Evaluate fitness Fi at current position Xi
* If Fi is better than Pbest-i then update Pbest-i and Pbest-i position marked as Pi
* If Fi is better than Gbest-i then update Gbest-i and Gbest-i position marked as Gi
* For each particle

- Update velocity Vi and position Xi using:



For each particle i:

- Xi is a vector denoting its position

- Vi is the vector denoting its velocity

- Fi denotes the fitness score of Xi

- Pi is the best position that it has found so far

- Pbest-i denotes the fitness of Pi

- Gi is the best position that has been found so far in its neighborhood

- Gbest-i denotes the fitness of Gi

Velocity update: - U(0,ϕi ) is a random vector uniformly distributed in [0,ϕi ] generated at each generation for each particle

- ϕ1 and ϕ2 are the acceleration coefficients determining the scale of the forces in the direction of pi and gi

- X denotes the element-wise multiplication operator

Island PSO algorithm :

* Divide particles into K groups where each group behave as an independent swarm.
* For each group Ki: randomly initialize particles positions and velocities
* While stopping criterion is not met do:
* For each group Ki :

- For each particle j that belongs to Ki:

- Evaluate fitness Fj at current position Xj.

- If Fj is better than Ki-Pbest-j then update Ki-Pbest-j and Ki-Pbest-j position Pj

- If Fj is better than Ki-Gbest-j then update Ki-Gbest-j and Ki-Gbest-j position marked as Ki-Gj

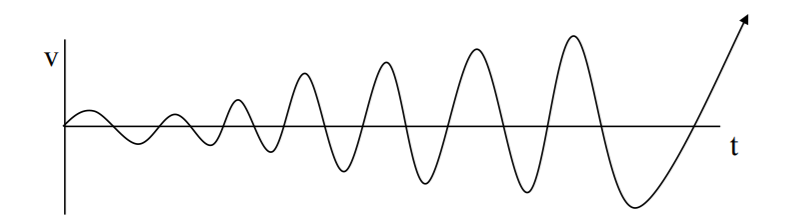
- For each particle j that belongs to Ki update Ki-Vj and Ki-Pj using the classic vector equation described above.

Acceleration coefficients - some useful insights :

* ϕ1>0, ϕ2 =0 particles are independent hill-climbers
* ϕ1 =0, ϕ2>0 swarm is one stochastic hill-climber
* ϕ1 =ϕ2>0 particles are attracted to the average of pi and gi
* Adaptive acceleration coefficients have also been proposed. For example to have ϕ1 and ϕ2 decreased over time

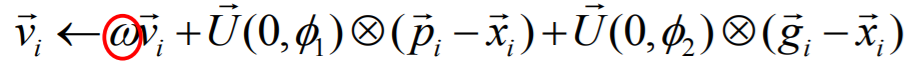
Original PSO problems:

* High acceleration coefficients result in less stable systems in which the velocity has the tendency to explode.
* To fix this, the velocity vi is usually kept within the range [Vmin , Vmax]
* However, limiting the velocity does not necessarily prevent particles from leaving the search space, nor does it help to guarantee convergence



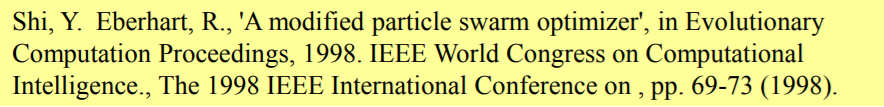
Velocity explosion solution:

An inertia weight ω was introduced to control the velocity explosion:



If ω, ϕ1 and ϕ2 are set correctly, this update rule allows for convergence without the use of Vmax.

Rule-of-thumb settings: ω = 0.7298 and ϕ1 = ϕ2 = 1.49618



**PSO Converging Islands process of work:**

**Motivation:**

Naive PSO can be good for local search which focus on one section of the search space.

Without any heuristic on the search space the section that will be explored is randomized dew to the random initialization.

Naive PSO is also good for global search when the cognitive weight is set high.

Island PSO can achieve better utilization of local search dew to the separation of the flocks.

Converging Island PSO tries to achieve **both global and local search at the same time.**

Based on the following assumption :

Start with Island PSO model when reaching to a certain amount of the total number of iteration, unify the Island that his G-best score is lowest with the Island who’s G-best score is highest, into one Island.

Particles from the low scored islands will migrate to the high score island and therefore will better utilize iterations to explore the “Hot Spot” founded.

Repeat the process until all islands converge into one big flock.

**Problems and difficulties**

During the migration particles can calculate position that been calculated by a member from their original Island (migrating in a row ), doing so wasting precious iterations.

The original design of the program had consideration to boids:

Boids consist of:

* Separation - try to move away from neighbors if these are to close.
* Alignment - steer towards the average heading of neighbors
* Cohesion - try to move toward the average position of neighbors



In the early design of my software i have implemented a graph data structure that hold particles.

Edges of the graph are weighted, and represents the distance between one particle to another.

If a particle restrict from the maximal or minimal distance it is relocated to a point that holds the restriction.

After testing and consulting Dr. Shir, we concluded that each iteration requires to re-calculate the distances of the edges of each graph.

This is computationally expensive, nevertheless when dealing with multidimensional non convex fitness functions, the probability of a point to be calculated twice is infinitesimal.

Therefore boid restriction in my software was excluded (though can be used).

In order to achieve information about terrains that are likely to have global optimum, i had to think about limiting each island boundaries.

If particles of each island are not limited, they would fly around the search space evaluating coordinates in a mixture with particles from other islands.

The goal was to find a way to divide N dimensional terrain into K equal parts.

This is a hard problem i encountered and yet found a solution.

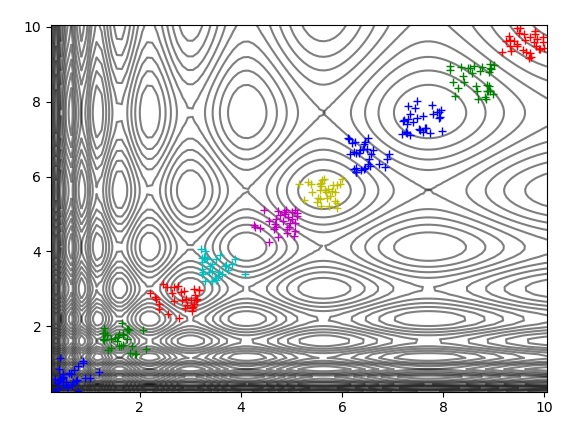
I overcame the described problem partially by using the following division:

Given the range where each variable is defined and the number of islands, perform a division of the search space :

Compute the length = (function upper bound - function lower bound ) / k

Where k is the number of islands.

The following picture represents a 2 dimensional partition of the terrains using the method described above:



I will refer to the partition showed above as - main diagonal initialization.

Another problem i encountered is finding the best configuration for heuristic parameters such as :

* Total number of islands.
* Unification parameters.
* Number of particles in each island
* Acceleration coefficient values.

And so on.

Finding the best configuration may take a long time due to the comparison method which based on a sufficiently high budget of operations, in order to take out the element of randomness.

The last problem i will mention is:

Determination if a global optima is unique.

When investigating multimodal fitness function in high dimensions, i worked with an epsilon approximation in order to determine if a particle reached global optima.

When more then one particle reached a global optima it was hard to say or calculate if the global optima found was unique.

The unique optima problem remains open.

**PSO - Converging Islands Model pseudo algorithm:**

* Divide particles into K groups where each group behaves as an independent swarm.
* For each group Ki: divide function terrain so particles are initialized within their boundaries.
* While stopping criterion is not met do:
* For each group Ki :
* Evaluate each particle in swarm Ki.
* Update swarm Gbest, particles Pbest and velocities.

- If unification criterion met:

- unify (Ki,Kj)

Where Ki is the Island whose G-best value is the highest among all islands.

And Kj is the Island whose G-best value is the lowest among all islands.

**Program implementation:**

My program is written in Python programing language.

I choose Python due to the basic experience i had from computational intelligence course.

Python is user friendly, popular and have many code packages.

I used Github as the version control software, Github enabled me to roll back safely to previous working versions of my program as well as following the development process.

My program consists from the following classes :

**Particle** - which represents a bird and has methods to evaluate his position, update his velocity and position.

**Island -** which represents a flock of birds and contains the class Particle.

**Space** - which represents the terrain on which all the birds flocks Space class contains the Island class.

**Main** - is only used to create an instance of Space object and pass the parameters needed.

Configuration files - on our meeting Dr.Shir guided me into creating configuration files that will ease passing of the parameters like acceleration coefficients, number of islands benchmark function details etc.

Graph library - implemented but not being used.

Benchmark functions - implementation of investigated benchmark functions.

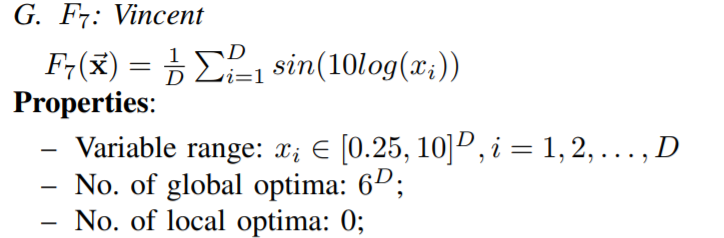
**Benchmark functions investigated :**

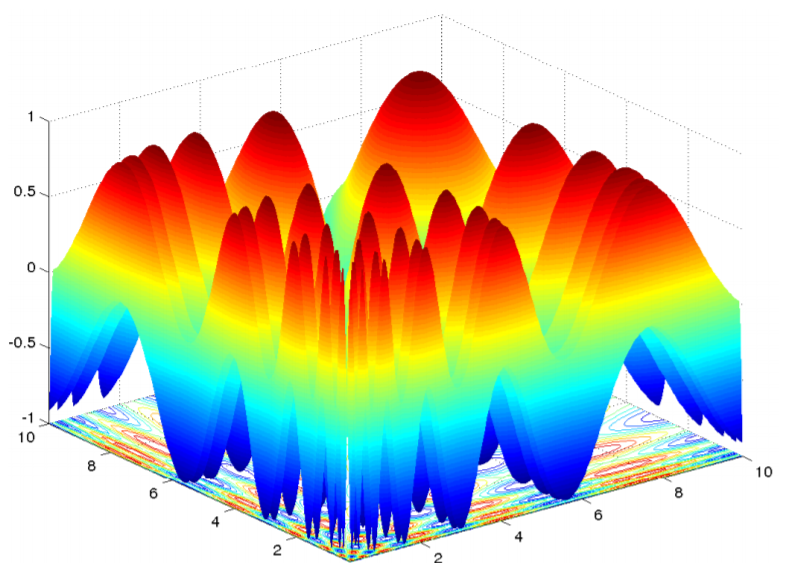
Source:

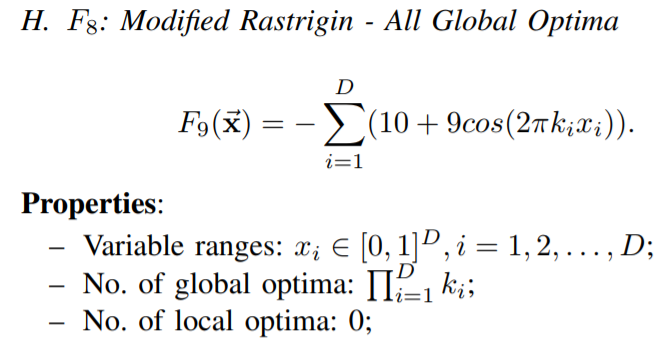
Benchmark Functions for CEC’2013 Special

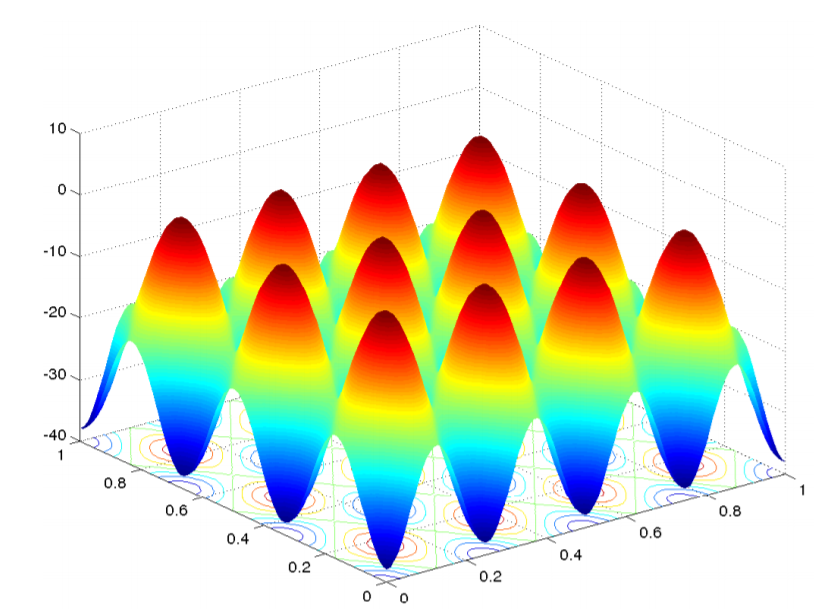
Session and Competition on Niching Methods for

Multimodal Function Optimization









**Initial testing:**

The process of testing a new optimization algorithm is long and require computational power which i don’t posses.

The testing of my algorithm executed on my personal machine, naturally the process took a long time.

In order to fully investigate the efficiency of converging islands approach we will have to compare results obtained from all of the fitness functions in the benchmark document.

**Test details:**

Reference method : naive PSO.

Benchmark functions: Vincent , Modified Rastrigin.

Dimensions: {10,30,80}.

Number of operations for each dimension: 30.

Number of particles : 300 (standard PSO), 30 particles in each 10 Islands (converging islands PSO).

Acceleration coefficients: set to 1.49 both social and personal in both methods tested.

Number of iterations: 100\*n\*n (where n represents the problem dimension).

Boundary condition: regular boundary condition (if a particle restrict from bound he sets his position to the bound ).

* Please note that Billiard boundary condition was implemented after testing.

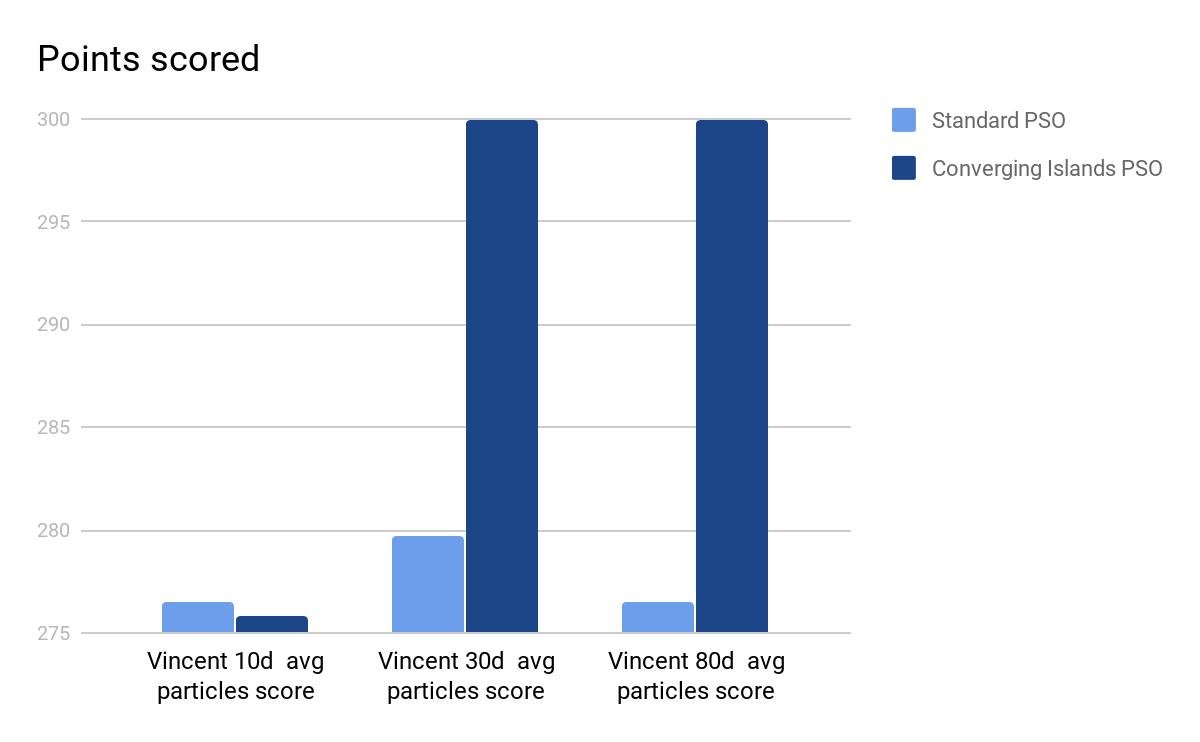
Epsilon approximation : 0.0001.

Initialization method: PSO - random , Converging Islands - main diagonal initialization.

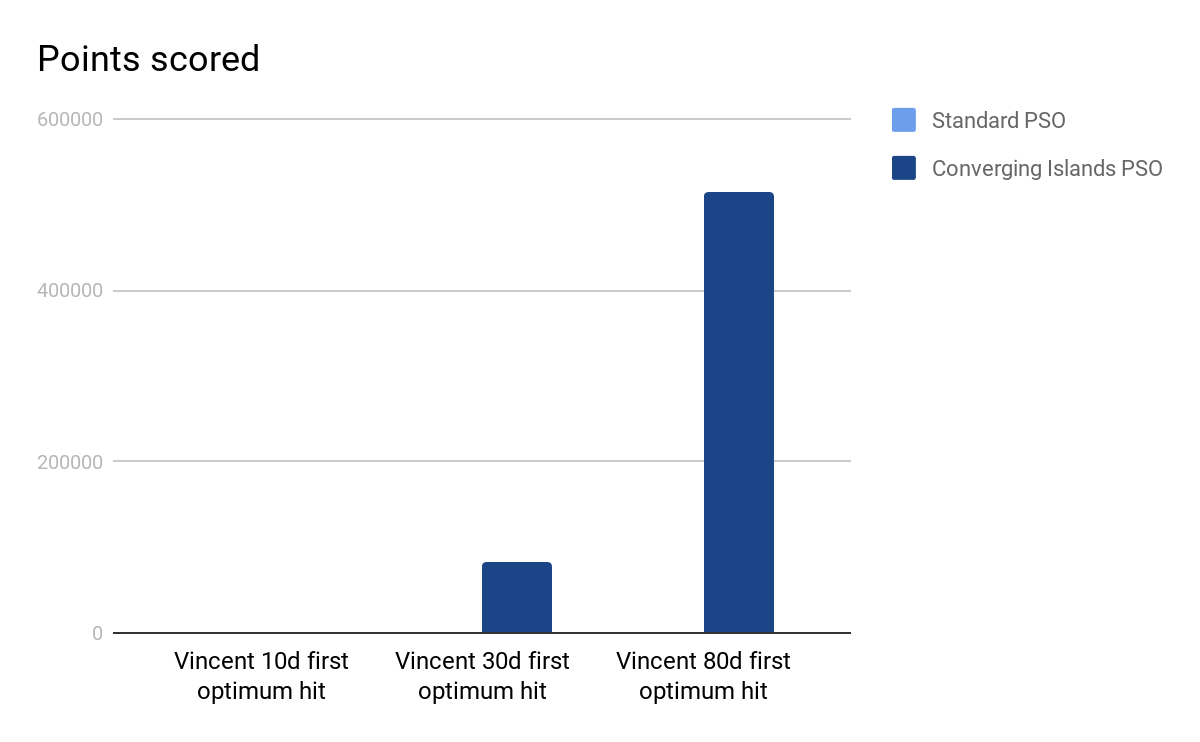
**Test results:**

We focused in two main parameters of success:

* Average particle scores on each operation (sum of all particles in the end of the iteration budget).
* Iteration number when first optimum hit occurs.



Where the maximum result is 300.



Please note that in 10d Vincent both algorithms did not hit any optima.

In 30d Vincent with Converging islands PSO all operations hitted optima, at average iteration 83105 where the max iteration budget for each operation is 90000.

In 80d Vincent with Converging islands PSO all operations hitted optima, at average iteration 515701 where the max iteration budget for each operation is 640000.

----------------------------------------------------------------------------------------------------------------

Modified Rastrigin benchmark function was sample tested and reached similar results.

----------------------------------------------------------------------------------------------------------------

**Testing initialization disclaimer:**

Testing with the main diagonal initialization has a heuristic characteristic, and therefore damages the meta - heuristic character of PSO.

In sample testing Converging Islands PSO with random initialization, the algorithm yield moderate improvement from his reference method.

Data is insufficient to be presented here.

**Conclusions:**

The converging islands PSO is an interesting ideathat aspires to mimic natural behavior in a more precise manner.

When implementing the algorithm it was interesting to see it behave as expected.

In higher dimensions in vincent function we can see convergence of particles yielding better average score, and global optima hits.

**Due to the heuristic character of the initialization and the lack of testing, it is too soon to determine its efficiency.**

As of now what we gained is a proof of concept.

**Acknowledgements :**

Dr. Ofer Shir - guidance and supervision throughout the whole project.

Mr. Hagai Barmatz - mathematical assistance.

Mr. Tal Abziz - initial software design assistance

**Bibliography:**

1. Spatially Structured Evolutionary Algorithms - Artificial Evolution in Space and Time by Marco Tomassini Publisher - Springer-Verlag Berlin Heidelberg hard cover ISBN - 978-3-540-24193-5.
2. Boids - Background and update by Craig Reynolds url : <http://www.red3d.com/cwr/boids/>.
3. An analysis of Sub-swarms in multi-swarm systems by - Antonio Bolufe Rohler and Stephen Chen url: <http://www.academia.edu/1276629/An_Analysis_of_Sub-swarms_in_Multi-swarm_Systems>
4. Seeking multiple solutions : an updated survey on niching methods and their applications Xiaodong Li, senior member, IEEE, Michael G.Epitropakis, Kalyanmoy Deb , Fellow, IEEE, and Andries Engelbrecht, Senior member, IEEE
5. Benchmark Functions for CEC’2013 Special Session and Competition on Niching Methods for Multimodal Function Optimization by Xiaodong Li , Andries Engelbrecht , Michael G.Epitropakis.